Лабораторна робота № 22

Тема. Використання абстрактних класів для обробки споріднених об’єктів у формах.

Мета. Формування вмінь і навиків створення абстрактних класів засобами C#. Закріплення вмінь і навиків наслідування класів, використання об'єктів, підпрограм, елементів керування. Застосування вмінь і навиків програмування алгоритмів лінійної, розгалуженої та циклічної структури.

Контрольні запитання.

1. Для чого призначені абстрактні класи?
2. Як описуються абстрактні методи та класи? Чи може абстрактний метод міститися в неабстрактному класі? Чому?
3. Яке зв’язування по замовчуванню використовується для абстрактних методів? Якому модифікатору це відповідає?
4. З яким модифікатором мають перевизначатися абстрактні методи в породжених класах? Чи обов’язково це робити?
5. Який механізм виконання абстрактних методів?

Завдання.

1. **Створіть новий додаток-форму. Розробіть у цій формі абстрактний клас двовимірної фігури та породжений від нього клас прямокутника.**

Опис цих класів може бути, наприклад, таким:

```
abstract class Figure2D
{   public abstract double area();
    public abstract double perimeter();
    public abstract void Info();
}


class Rectangle : Figure2D
{   private double a, b;

    public double A
    {   get { return a; }
        set { if (value >= 0)
                a = value; } }

    public double B
    { get { return b; }
      set { if (value >= 0)
              b = value; } }

    public Rectangle(double a, double b)
    {   this.a = a; this.b = b; }

    public sealed override double area()
    {   return a * b; }

    public sealed override double perimeter()
    {   return 2 * (a + b); }

    public override void Info()
    {  MessageBox.Show("Дані прямокутника:\nдві сторони по " + a.ToString() + " та дві по " + b.ToString() +
                       " од.;\nплоща: " + area().ToString() + " кв. од.;\nпериметр: " + perimeter().ToString() +
                       " од.", "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information); }
}
```

2. **Опишіть абстрактний клас зображуваної тривимірної фігури з основою-прямокутником, реалізуючи наслідування від попереднього розробленого класу прямокутника. Створіть також у цьому класі поля і властивості для встановлення кольору фігури та абстрактний метод для зображення об’єкта, починаючи від заданої точки. Забезпечте наслідування породжених класів тривимірних фігур від цього абстрактного класу.**

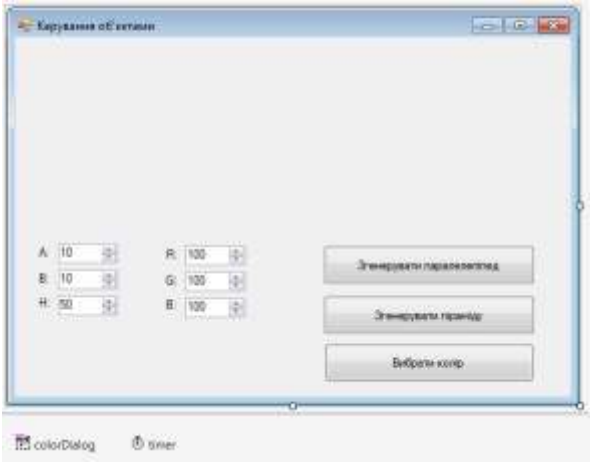
Наприклад, опис базового класу зображуваної тривимірної фігури з основою-прямокутником та породжених від цієї тривимірної фігури класів прямокутного паралелепіпеда і піраміди з основою-прямокутником буде таким:

```
abstract class DrawFigure3DBaseRectangle : Rectangle
 {private byte cr, cg, cb;
  public byte CR
  {  get { return cr; }
     set { if (value >= 0 && value <= 255)
             cr = value; } }

  public byte CG
  {  get { return cg; }
     set { if (value >= 0 && value <= 255)
             cg = value; } }

  public byte CB
  {  get { return cb; }
     set { if (value >= 0 && value <= 255)
             cb = value; } }

  public abstract void draw3D(int startX, int startY, PaintEventArgs e);
  private double h;

  public double H
  {get { return h; }
   set { if (value >= 0) h = value; }}

  public DrawFigure3DBaseRectangle(double a, double b, double h):base(a,b)
  {H = h;
  }

  public DrawFigure3DBaseRectangle(double a, double b, double h, byte cr, byte cg, byte cb):base(a,b)
  {H = h;
   CR = cr; CG = cg; CB = cb; }

  public double areaBase()
  {return base.area(); }

  public double perimeterBase()
  {return base.perimeter(); }

  public abstract double volume();
 }

class DrawParallelepiped : Figure3DBaseRectangle
 {public DrawParallelepiped(double a, double b, double h, byte cr, byte cg, byte cb) :
                              base(a, b, h, cr, cg, cb) { }

  public override void draw3D(int startX, int startY, PaintEventArgs e)
  {Pen pen = new Pen(Color.FromArgb(255, CR, CG, CB), 3);
   int x = (int)A, y = (int)B, z = (int)H;
   e.Graphics.DrawLine(pen, startX, startY, startX + x, startY);
   e.Graphics.DrawLine(pen, startX, startY, startX + y, startY + y);
   e.Graphics.DrawLine(pen, startX + y, startY + y, startX + y + x, startY + y);
   e.Graphics.DrawLine(pen, startX + x, startY, startX + y + x, startY + y);

   e.Graphics.DrawLine(pen, startX, startY, startX, startY + z);
   e.Graphics.DrawLine(pen, startX + x, startY, startX + x, startY + z);
   e.Graphics.DrawLine(pen, startX + y, startY + y, startX + y, startY + y + z);
   e.Graphics.DrawLine(pen, startX + x + y, startY + y, startX+x+y, startY+y+z);

   e.Graphics.DrawLine(pen, startX, startY + z, startX + x, startY + z);
   e.Graphics.DrawLine(pen, startX, startY + z, startX + y, startY + y + z);
   e.Graphics.DrawLine(pen, startX + y, startY + y + z, startX+y+x, startY+y+z);
   e.Graphics.DrawLine(pen, startX + x, startY + z, startX+y+x, startY+y+z);
  }

  public override double volume()
  {return A * B * H; }

  public override void Info()
  {MessageBox.Show("Дані паралелепіпеда:\nсторони по " + A.ToString() + ", " +
    B.ToString() + ", " + H.ToString() + " од.;\nплоща основи: " +
    area().ToString() + " кв. од.\nоб'єм: " + volume().ToString() + " куб. од.",
    "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information); }
}
```

```
class DrawPiramida : DrawFigure3DBaseRectangle
{
    public DrawPiramida(double a, double b, double h, byte cr, byte cg, byte cb) :
                    base(a, b, h, cr, cg, cb)
    {}

    public override void draw3D(int startX, int startY, PaintEventArgs e)
    {   Pen pen = new Pen(Color.FromArgb(255, CR, CG, CB), 3);
        int x = (int)A, y = (int)B, z = (int)H, vx = startX + (x + y) / 2, vy = startY + y / 2;
        e.Graphics.DrawLine(pen, vx, vy, startX, startY + z);
        e.Graphics.DrawLine(pen, vx, vy, startX + y, startY + y + z);
        e.Graphics.DrawLine(pen, vx, vy, startX + x, startY + z);
        e.Graphics.DrawLine(pen, vx, vy, startX + x + y, startY + y + z);

        e.Graphics.DrawLine(pen, startX, startY + z, startX + x, startY + z);
        e.Graphics.DrawLine(pen, startX, startY + z, startX + y, startY + y + z);
        e.Graphics.DrawLine(pen, startX + y, startY + y + z, startX + y + x, startY + y + z);
        e.Graphics.DrawLine(pen, startX + x, startY + z, startX + y + x, startY + y + z);
    }

    public override double volume()
    { return 1.0/3 * A * B * H; }

    public override void Info()
    { MessageBox.Show("Дані піраміди:\nсторони основи по " + A.ToString() + " та " +
                      B.ToString() + " од., висота: " + H.ToString() + " од.;\nплоща основи: " +
                      areaBase().ToString() + " кв. од.\nоб'єм: " + volume().ToString() +
                      " куб. од.", "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information); }
}
```

3. За допомогою списків об’єктів абстрактного класу зображуваної тривимірної фігури забезпечте створення та переміщення декількох зображень породжених об’єктів з головної кнопкової форми.

Наприклад, головна кнопкова форма з додатковими елементами керування, нумераторами і кнопками для створення геометричних фігур буде така:



А її код буде, наприклад, таким:

```
public partial class Form1 : Form
{   List<DrawFigure3DBaseRectangle> ListDrawFigure3D = new List<DrawFigure3DBaseRectangle>();

    public Form1()
    {   InitializeComponent();}

    private void button3_Click(object sender, EventArgs e)
    {
        var pd = new DrawParallelepiped((double)A.Value, (double)B.Value, (double)H.Value,
                                        (byte)CR.Value, (byte)CG.Value, (byte)CB.Value);
        pd.Info();
        ListDrawFigure3D.Add(pd);
        Refresh(); }

    private void button2_Click(object sender, EventArgs e)
    {   var pa = new DrawPiramida((double)A.Value, (double)B.Value, (double)H.Value,
                                  (byte)CR.Value, (byte)CG.Value, (byte)CB.Value);
        pa.Info();
        ListDrawFigure3D.Add(pa);
        Refresh(); }

    private void Form1_Paint(object sender, PaintEventArgs e)
    {int pozX = 10;
     foreach (DrawFigure3DBaseRectangle Figure3D in ListDrawFigure3D)
        {Figure3D.draw3D(DateTime.Now.Second + pozX, 10, e);
         pozX += 100; } }
```

```
    private void button1_Click(object sender, EventArgs e)
    { colorDialog.Color=Color.FromArgb((int)CR.Value, (int)CG.Value, (int)CB.Value);
      if (colorDialog.ShowDialog()==DialogResult.OK)
      {CR.Value = colorDialog.Color.R;
       CG.Value = colorDialog.Color.G;
       CB.Value = colorDialog.Color.B;
      } }

    private void timer_Tick(object sender, EventArgs e)
    { Refresh(); }
}
```